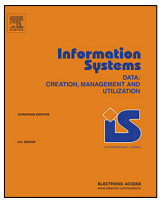
Information Systems 105 (2022) 101938

Contents lists available at ScienceDirect

Information Systems

journal homepage: www.elsevier.com/locate/is



LILLIE: Information extraction and database integration using linguistics and learning-based algorithms

Ellery Smith [a,*], Dimitris Papadopoulos [b], Martin Braschler [a], Kurt Stockinger [a]

[a] Zurich University of Applied Sciences, Switzerland
[b] Technical University of Crete, Greece

ARTICLE INFO

Article history:
Received 25 October 2021
Received in revised form 2 November 2021
Accepted 3 November 2021
Available online 18 November 2021
Recommended by Dennis Shasha

Keywords:
Information extraction
Data integration
Machine learning for database systems

ABSTRACT

Querying both structured and unstructured data via a single common query interface such as SQL or natural language has been a long standing research goal. Moreover, as methods for extracting information from unstructured data become ever more powerful, the desire to integrate the output of such extraction processes with "clean", structured data grows. We are convinced that for successful integration into databases, such extracted information in the form of "triples" needs to be both (1) of high quality and (2) have the necessary generality to link up with varying forms of structured data. It is the combination of *both* these aspects, which heretofore have been usually treated in isolation, where our approach breaks new ground.

The cornerstone of our work is a novel, generic method for extracting open information triples from unstructured text, using a combination of *linguistics and learning-based extraction* methods, thus uniquely balancing both precision and recall. Our system called LILLIE (LInked Linguistics and Learning-Based Information Extractor) uses *dependency tree modification rules* to refine triples from a high-recall learning-based engine, and combines them with syntactic triples from a high-precision engine to increase effectiveness. In addition, our system features several augmentations, which modify the generality and the degree of granularity of the output triples. Even though our focus is on addressing both quality and generality simultaneously, our new method substantially outperforms current state-of-the-art systems on the two widely-used CaRB and Re-OIE16 benchmark sets for information extraction.

We have made our code publicly available[1] to facilitate further research.

1. Introduction

It is commonly known that some 80% of enterprise data is unstructured while only some 20% is structured [1,2]. Hence, only a relatively small part of enterprise data is stored in relational databases. Traditionally, the information retrieval and natural language processing communities focus on working with unstructured data, while the database community typically works with structured data. In order to query both structured and unstructured data via a single common query interface such as SQL or natural language [3,4], there have been several research efforts over the last years. One such approach, which we follow in our work, is to first use *information extraction* techniques to retrieve relevant entities (subjects and objects) and relationships (predicates) from text documents and to then populate so-called *knowledge graphs* or *ontologies* [5]. The next step is to link the generated knowledge graph with the tables of a relational database (*entity linking*). Finally, the combined system can be queried in either SQL or in natural language.

An example of such an end-to-end data processing pipeline is shown in Fig. 1. The inputs are *text documents* from medical articles such as in PubMed,[2] as well as a *relational database* that stores information about genes and so-called anatomical entities, i.e. different organs in our body. First, the input text "*THY1 is overexpressed in human gallbladder carcinoma*" is parsed and subject, object and predicate are extracted. Next, the subject "THY1" and the object "human gallbladder carcinoma" are linked to the relational database.

Building such an end-to-end pipeline to enable the vision of querying structured and unstructured data via a common interface has been a long standing research effort [6,7]. However, each of the previously-mentioned steps has typically been treated and

* Corresponding author.
E-mail addresses: ellery.smith@zhaw.ch (E. Smith), dpapadopoulos@infili.com (D. Papadopoulos), martin.braschler@zhaw.ch (M. Braschler), kurt.stockinger@zhaw.ch (K. Stockinger).

[1] https://github.com/OIELILLIE/LILLIE

[2] https://pubmed.ncbi.nlm.nih.gov/.

https://doi.org/10.1016/j.is.2021.101938

Input text:
THY1 is overexpressed in human gallbladder carcinoma.

Extracted Triple:
- **Subject:**
  - Text: `THY1`
  - Linked Entity: *gene:thy1*
- **Predicate:** `is overexpressed in`
- **Object:**
  - Text: `human gallbladder carcinoma`
  - Linked Entity: *uberon:0002110*

Ontology-linked Triple:
gene:thy1 ; `is overexpressed in` ; *uberon:0002110*

Fig. 1. Example of an end-to-end data processing pipeline. Step 1 - *Information Extraction*: First we extract triples such as subject, predicate and object from a text document. Step 2 - *Entity Linking*: Afterwards we link the extracted subjects and objects to specific columns of a relational database. As a result we have an extended relational database that is enriched with information stored in text documents.

optimized in isolation. Hence, significant potential for improvement is left unexplored when considered in the larger context of data exploration. When the triple extraction process is viewed as an integral part of this larger process of integrating and querying structured and unstructured data, we claim that two considerations are crucial to be treated simultaneously — the combination of which has previously received only little attention:

- How to *optimize the effectiveness* of triple extraction, balancing both recall and precision?
- How to augment the approach to *increase generality*, making the extracted triples suitable for linking up with varying forms of structured data stored in a relational database?

Much work on triple extraction concentrates on the first aspect only and is therefore not really optimized towards an end-to-end pipeline of both triple extraction and database integration.

In this work we tackle an important open gap in triple extraction and database integration. In particular, we present a novel approach for extracting subject–predicate–object relational triples from unstructured text, which are then linked to relevant ontologies and inserted into a relational database. This paper is part of a greater vision of building a data exploration system with INODE [8].

The *contributions of our paper* are as follows:

- We combine a *high-precision rule-based* triple extractor with a *high-recall learning-based* extractor, using a novel triple refinement method.
- Our system includes additional options for *output modifications*, which allow the granularity and specificity of the extracted triples to be *customized to a given structured database*.
- Our approach *outperforms current state of the art* systems on the two widely-used benchmark datasets CaRB and ReOIE.

The paper is organized as follows: in Section 2 we review the related work on information extraction and entity linking for knowledge base construction; in Section 3, we give an overview of the LILLIE architecture; in Sections 4 and 5, we describe the algorithms and functions of the rule-based extractor and the learning-based extractor, respectively; in Sections 6 and 7, we show how to combine both engines, and customize their output; in Section 8, we describe how to apply our triple extractor to the task of entity linking and database insertion; in Section 9, we give a detailed analysis and evaluation of all the components of our system, and compare these to the current state-of-the-art. The paper culminates in Section 10, where we show how the enriched database can be queried and discuss performance considerations.

2. Related work

In this section, we provide background information for each of the distinctive modules that comprise our data processing pipeline, namely open information extraction and entity linking for knowledge base construction.

2.1. Information extraction

Information extraction systems aim at distilling structured representations of information from natural language text, usually in the form of relational triples *{subject, predicate, object}*, which correspond to *{entity1; relationship; entity2}* or n-ary propositions [9].

There are two types of information extraction systems: *Closed Information Extraction (CIE)* systems identify instances from a fixed and finite set of corpora, considering only a closed set of relationships between two arguments [10]. On the other hand, *Open Information Extraction (OIE)* systems use a domain-independent approach and are capable of extracting entities and relationship triples from natural language sentences. Since OIE systems follow a relation-independent extraction paradigm, they can play a key role in many natural language processing (NLP) applications involving natural understanding (NLU) and knowledge base construction from massive and heterogeneous corpora, by extracting phrases that indicate semantic relationships between entities.

In order to extract triples, most approaches try to identify linguistic extraction patterns, either hand-crafted or automatically learned from the data. The line of work on OIE starts with systems relying on distant supervision [11,12], and rule-based paradigms that focus on the grammatical and syntactic properties of the language [13,14]. An abundance of learning-based systems that leverage annotated data sources to train classifiers has been proposed [15,16], with more recent implementations making use of pretrained language models [17,18]. Despite the existence of so many approaches, however, the majority focus only on evaluating the effectiveness of different triple extraction tools on raw data, without incorporating any preprocessing strategies to limit the number of potentially uninformative triples [19].

Some more recent methods go beyond the triple extraction task by encompassing more thorough preprocessing and postprocessing strategies, including discourse analysis, coreference resolution or summarization to improve the quality of the extracted triples [20–22].

2.2. Entity Linking for knowledge base construction

Entity Linking (EL) – also known as *Named Entity Recognition / Disambiguation* – is the task of identifying an entity mention in a text and establishing a link to an entry in a knowledge base (KB), e.g. Wikidata [23], DBpedia [24], YAGO [25]. EL systems are capable of resolving the lexical ambiguity of entity mentions and can therefore be extremely useful in a plethora of NLU applications, by enriching the information extracted via OIE systems. Moreover, by establishing links between the entity mentions and KB entities, we are able to store and utilize information in semantic graphs, facilitating semantic parsing, question answering and exploratory data analysis operations.

Earlier approaches leverage statistical models combined with feature engineering methods to achieve entity linking, viewing the problem as a word sequence labeling task [26]. More modern neural-based approaches treat the problem as a multi-class classification task, in which entities correspond to classes. The goal is to propose a list of candidate entities for each mention by encoding both the mentions and the candidate entities into vector representations, then ranking the candidates based on

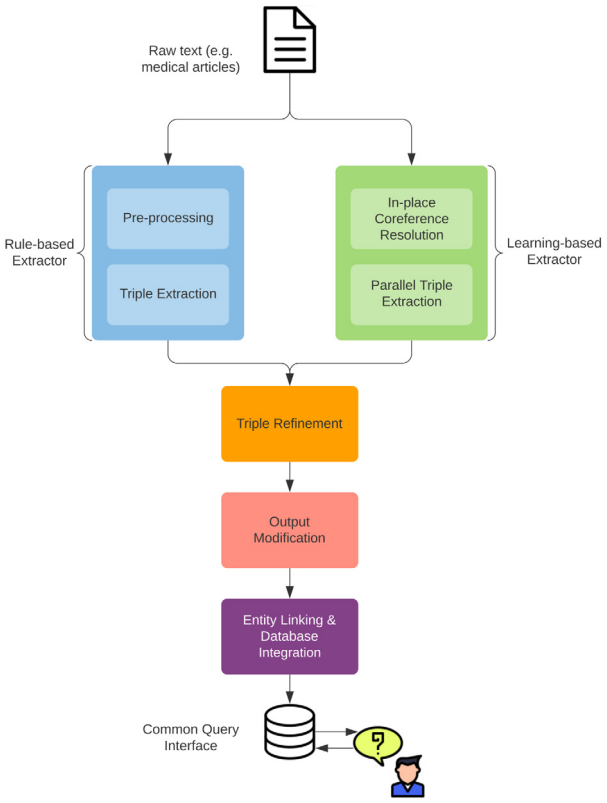


Fig. 2. Overview of our proposed architecture of LILLIE for extracting relational triples from text documents and integrating them into a relational database. Uniquely, our system is a combination of a rule-based extractor (high precision-oriented) with a learning-based extractor (high recall-oriented).

content similarity. Much work has been put in constructing and correlating mention and candidate entity embeddings, spanning from convolutional encoders [27,28] to recurrent [29,30] and self-attention networks [31–35].

3. Architecture of LILLIE

An overview of the architecture of LILLIE (LInked Linguistics and Learning-Based Information Extractor) is shown in Fig. 2. We briefly describe the main aspects of each of the components. The details will be discussed in the subsequent sections.

- *Rule-based Extractor*: This component follows a precision-oriented, linguistics-based approach to extract triples from unstructured text (see Section 4).
- *Learning-based Extractor*: This component follows a recall-oriented triple extraction approach, based on complementary OIE strategies to extract relational triples (see Section 5).
- *Triple Refinement*: This module efficiently combines the results of the aforementioned extractors, maintaining the best attributes of each one (see Section 6).
- *Output Modification*: A number of parameterization settings are introduced by this module, allowing LILLIE to adapt to different text domains (see Section 7).
- *Entity Linking and Database Integration*: The final part of our system aims at correlating the extracted triples with domain-specific ontologies in order to enhance their contextual value, before integrating them to a relational database (see Section 8).

4. The rule-based extractor

In this section we explain our *precision-oriented, rule-based* extractor, whose goal is to extract relational triples from a text document.

The input for the rule-based extractor is a sentence of unstructured plaintext, such as:

> Long non-coding RNA CCAT2 promotes breast cancer growth and metastasis

And the output is a set of annotated subject–predicate–object triples, for example:

```
(long non-coding RNA CCAT2 ; promotes ; breast cancer growth)

(long non-coding RNA CCAT2 ; promotes ; breast cancer metastasis)
```

For each component of a triple (subject, predicate and object), we identify a single base term from the input text. From this term, we expand it into a complete phrase, while annotating each term with the rules used to include it.

We explain this concept by using the previous example:

```
long non-coding RNA CCAT2 ; promotes ; breast cancer growth
```

Here, `long non-coding` is marked as an *adjectival component* of the subject, `RNA` is marked as a *compound element*, and `CCAT2` is marked as the *base term*. With this additional information, we can alter the granularity of the given triples, for matching to more or less specific entities in structured data. For example, if `"long non-coding RNA CCAT2"` is not present in an ontology, we can match on the more general entities such as `"RNA CCAT2"` or `"CCAT2"`.

We design our rules to be as generic as possible, and to be applicable on all possible text domains. To this end, we used the CaRB and ReOIE16 datasets for crafting these rules. The CaRB [36] and ReOIE16 [37] benchmarking sets are 1877 arbitrarily selected annotated sentences from various sources, designed to represent a wide variety of text domains and language styles.

To design the rules, we analyzed a set of only 30 random sentences from the CaRB development set, and designed the rules based on these, using an approach similar to a "few shot" learning model. We then tested these rules on the rest of the CaRB development set to assess their generalizability. This approach ensures that our system is generic, and adaptable to many different domains, as demonstrated by its performance on the ReOIE16 set, which was not seen prior to evaluation.

We believe these rules to be applicable to a wide variety of textual domains, with no modifications to the core rule set being needed as we move between datasets. In Section 7, we describe a small number of domain-specific adaptions layered on top, that can be enabled or disabled when required, to give further generalizability. This allows for high-precision annotated extractions, suitable for to mapping to structured data.

4.1. Pre-processing

To begin with, the input text is parsed into a *syntactic dependency tree* using the Stanford Dependency Parser [38], and annotated with part-of-speech tags. The dependency tree passes through several custom pre-processing stages, before the tokens comprising the triple are extracted from the tree.

We include a purpose-built *anaphora resolution procedure* in our system, based on deterministic transformation rules on the dependency tree. We keep the ruleset minimal, in order to

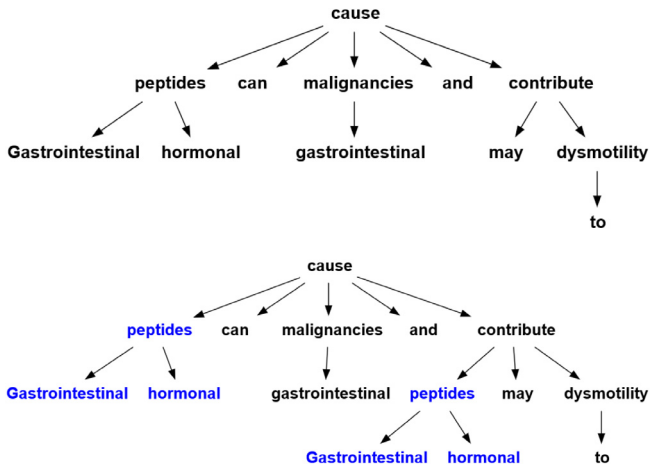


Fig. 3. A syntactic dependency tree before pre-processing (above) and after (below). The subject-phrase "`Gastrointestinal hormonal peptides`", shown in blue, has been duplicated in the sub-clause on the left, to account for the conjunctive phrase. With this pre-processing step, LILLIE enables more effective downstream processing. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

achieve a reliable, high-precision procedure, in contrast to generalized anaphora resolution systems, which we found introduced many noisy substitutions. In addition, we account for conjunctive predicates by duplicating and re-arranging branches of the dependency tree.

An example of this pre-processing is shown in Fig. 3 for the sentence *"Gastrointestinal hormonal peptides can cause gastrointestinal malignancies and may contribute to dysmotility"*.

This procedure handles cases of co-reference resolution where the anaphora is a "wh-word" ('which' or 'who'), or a pronominal (it, he, etc.), at the head of a sub-clause. These cases can be deterministically resolved, e.g. "This cancer is found in the kidneys, where it can be aggressive.". These instances can to be reliably resolved with very high precision, as they are linguistically unambiguous. Cases where: a) there are two (or more) "wh-words" or pronominals in the dependent clause; or b) where the anaphora is not the head-word of a sub-clause; are explicitly not handled, as they can be ambiguously interpreted, and attempting to resolve them would break the principle of high precision extraction.

Similarly, verbal conjunctions, such as in "He was the Prime Minister and also served as a judge." (from the CaRB development set), are handled only in cases where the subject can be unambiguously added to the conjunctive clause, but will not be split in more complex cases, such as "This gene is found in the kidney and it mutates within it.". Such cases are deferred to the Learning-Based Extractor — which defines our complementary approach. We provide an analysis of how much this approach (of not attempting to resolve ambiguity) improves F1 and AUC in Table 6.

The procedure is described formally in Algorithm 1. We use $d(n, n')$ to denote the syntactic dependency between nodes n and n' of the dependency tree, where n' is the governor. Further information concerning the syntactic and linguistic terms used in the algorithms in this work can be found in [39].

4.2. Triple extraction

After processing the original dependency tree into a set of modified trees, we can now use it to extract *relational triples*. To do this, we find sets of three nodes from each tree, which represent the base terms for the subject, predicate and object, respectively. We again explain this process with our running example:

Algorithm 1 Input Pre-processing

Input: t, a syntactic dependency tree
Output: T, a set of modified trees

for all $n \in$ BreadthFirstSearch(t) **do**
  for all $c \in n.children$ **do**
    if $d(c, n) = conj$ **and** n is verbal **then**
      for all $c' \in n.children$ **do**
        if $d(c', n) = nsubj$ **then**
          $c.children = c.children \cup \{\text{Copy}(s)\}$
$T \leftarrow \{t\}$
for all $n \in$ BreadthFirstSearch(t) **do**
  if $d(n, n.parent) \in \{acl, aclrelcl, advcl\}$ **then**
    for all $c \in n.children$ **do**
      if $d(c, n)$ is a subject **then**
        if c is a *wh*-word or pronominal **then**
          $c \leftarrow n.parent.subject$
    $t' \leftarrow$ CopyTree(n); $T \leftarrow T \cup \{t'\}$

Long non-coding RNA CCAT2 promotes breast cancer growth and metastasis

The base tokens are `CCAT2` and `promotes`, for the subjects and predicates respectively, and `metastasis` and `growth` for each of the two objects. These are identified according to a set of rules, which determine the appropriate dependencies between the three terms. The edges between the subject, predicate and object base tokens must match a set of valid edges, such as (*nsubj*→*dobj*) for a nominal subject and direct object triple.

For each of these base terms, we traverse the dependency tree depth-first, marking terms that match certain rules. These rules take the form of a lookup table, where the dependency on each edge is mapped to six rules: three *inclusion rules*, one for subject, predicate and object: and, similarly, three *donation rules*. Depending on whether the base token is for the subject, predicate or object, two of these rules are then chosen. The *inclusion rules* determine whether an edge is valid to traverse, and the sub-tree can be included in the final triple. The *donation rules* determine if a given sub-tree will be donated, or moved, to the subject, predicate or object.

For example, a *compound* edge below the subject node will be included in the subject portion of the triple. If this edge was below the predicate node, however, it would be moved to the object's subtree. This process is applied recursively to all subtrees until the subject, predicate and object trees contain only relevant tokens.

For example, an auxiliary verb, such as *may* or *could*, will be annotated and added to the predicate, and compound nouns, such as *breast* and *cancer* will be added to the subject or object. The output of this procedure is a set of triples, consisting of terms annotated with dependency rules.

In total, there are 40 branch-rules used in our system. Here, we give an example of some of these rules:

- **Temporal Modifiers**: A temporal modifier denotes the time at which the predicate was invoked by the subject on the object. In the sentence "Last year, he passed his exams.", *last year* is the temporal modifier.
  Temporal phrases are typically attached to the verb of a clause in a dependency graph; however, we transfer all instances of temporal indicators to the object portion of the triple using the branch donation rules. In the previous example, we take *passed* as the predicate, and transfer the phrase *last year* to the object, giving the triple (`he ; passed`

Table 1
A sample of the branch-rules used in our rule-based extractor. *S*, *P* and *O* represent Subject, Predicate and Object, respectively.

Rule	Remove from S	Remove from P	Remove from O	Donate from S to	Donate from P to	Donate from O to
Temporal modifiers	No	Yes	No	N/A	O	N/A
Adjectival clauses	No	Yes	No	S	N/A	O
Compound particle	Yes	No	Yes	N/A	P	N/A
Multi-word expressions	No	Yes	No	S	N/A	O
Copulas	Yes	No	No	N/A	P	P
...	...	...	...	...	...	...

`; his exams last year)`. This differs from the standard parse of `(he ; passed last year ; his exams)`, and produces a more informative triple for structured database insertion by keeping specificity of the predicate high.

- **Compound Particles**: A compound particle denotes part of an idiomatic split verbal phrase, such as in the sentence "The detective closed the case down.", where *down* is a compound particle in the verb phrase *shut down*.
  Using this rule, we can combine verb phrases split over long distances, and remove ambiguity in the object, by using dependency tree analysis. In the previous example, we can extract the triple `(the detective ; closed down ; the case)`, rather than the more ambiguous interpretation: `(the detective ; closed ; the case down)`, accounting for the clear semantic difference between the two predicates.
- **Multi-Word Expressions**: Multi-word expressions (MWEs) cover phrases like *such as* or *instead of*. In the sentence, "We could find a suitable donor, at least.", the phrase *at least* is attached as a MWE to the root verb.
  In cases such as these, where the MWE will cause ambiguity in the predicate, even though it is a dependent branch of the root verb, we use the inclusion rules to prune its branch from the tree, to give the triple `(we ; could find ; a suitable donor)`, rather than the direct parses of `(we ; could at least find ; a suitable donor)`. In the subject and object, however, we leave MWEs in the output triple, as they frequently contribute necessary context in noun phrases.
- **Copulas and Auxiliary Verbs**: A copula generally denotes an *is-a* relation, as in "CCAT2 is a gene" or "These tumours were malignant." Auxiliary verbs occur in phrases such as "was found" (*was*) or "has been detected" (*has* and *been*).
  In a typical dependency parse, copulas are typically not positioned as the root of the tree, and are instead represented as a sub-branch of the object. As such, we modify the tree, by promoting the copula to the root, and positioning the object as a sub-tree of the root verb. However, in cases where copulas and auxiliary verbs interact, as in "These tumours have been malignant.", we use our rules to ensure that the extracted triple becomes `(these tumours ; have been ; malignant)`, rather than `(these tumours ; have ; been malignant)`, as the dependency tree suggests.

With these rules, we attempt to cover all linguistic domains, but, similarly to our pre-processing techniques, we take the approach of removing portions of the input that are linguistically ambiguous, and thus benefit from a stochastic approach, as these will be handed by the Learning-Based Extractor. As such, some dependencies, such as *Clausal Subjects* (e.g. in "How this gene behaves makes little sense."), or *Discourse* (representing elements of casual speech) are pruned from the dependency tree by our rules. We implement these rules using a table, describing when to include, remove, or transfer branches below the subject, predicate or object base token. An abridged version of this table is shown in Table 1.

5. The learning-based extractor

The learning-based extractor introduces a data processing pipeline that takes as input a sentence of natural language text and provides a structured representation of the extracted information in the form of OIE triples, identical to the rule-based extractor. For example, the same sentence:

> Long non-coding RNA CCAT2 promotes breast cancer growth and metastasis

Produces the following triple in the form of subject–predicate–object:

```
(long non-coding RNA CCAT2 ; promotes ;
 breast cancer growth and metastasis)
```

The extractor comprises an *in-place coreference resolution* module and a *parallel triple extraction module* that integrates three complementary OIE engines relying on both learning-based and linguistics-based components, each based on a different extraction strategy. These engines are discussed in detail in Section 5.2. The main intuition behind this approach is to enhance the performance of our extractor compared to standalone engines, while developing a recall-oriented approach to be used in conjunction with the precision-oriented rule-based extractor. More information regarding the aforementioned modules is provided in the following subsections.

5.1. In-place coreference resolution

An in-place coreference resolution module is used to improve the quality of information extraction, addressing those cases where an entity found in unstructured text is replaced by its coreferential pronoun. For example, the phrase "Mary is a nice person, I like hanging out with *her*." will be substituted with the coreference-resolved equivalent "Mary is a nice person, I like hanging out with *Mary*."

To this end, we leverage the *pretrained neural coreference resolution* tool from AllenNLP [40], which implements a variant of the Lee et al. end-to-end coreference resolution model [41] using Span-BERT embeddings [42]. The model has been trained on the OntoNotes 5.0 dataset [43], achieving an F1-score of 78.87% on the test set. Prior to being ingested by our triple extraction module, each sentence is processed through the in-place coreference resolution component, where all mentions referring to the same entity are substituted by that entity, eventually leading to more informative triples.

An indicative example of the performed substitutions on an small extract is provided in Table 2. As shown in the example, all mentions of the type *"this gene"* have been replaced with the original entity *"CCL26 gene"*, while all mentions of the type *"this chemokine"* have been substituted with the original entity *"chemokine receptor CCR3"*. It is evident that the coreference-resolved text will lead to more informative triples, since all triple arguments will contain original noun phrases as unique referents instead of their coreferential pronouns.

Table 2
In-place coreference resolution using the learning-based extractor on an example taken from the OncoMX gene biomarkers database. LILLIE replaces all mentions of "this gene" with the original entity "CCL26 gene".

Original sentences:
CCL26: **This gene** is one of two Cys–Cys (CC) cytokine genes clustered on the q arm of chromosome. Cytokines are a family of secreted proteins involved in immunoregulatory and inflammatory processes. The CC cytokines are proteins characterized by two adjacentcysteines. The cytokine encoded by **this gene** displays chemotactic activity for normal peripheral blood eosinophils and basophils. The product of **this gene** is one of three related chemokines that specifically activate chemokine receptor CCR3. **This chemokine** may contribute to the eosinophil accumulation in atopic diseases.
Coreference-resolved sentences:
CCL26: **CCL26 gene** is one of two Cys–Cys (CC) cytokine genes clustered on the q arm of chromosome. Cytokines are a family of secreted proteins involved in immunoregulatory and inflammatory processes. The CC cytokines are proteins characterized by two adjacent cysteines. The cytokine encoded by **CCL26 gene** displays chemotactic activity for normal peripheral blood eosinophils and basophils. The product of **CCL26 gene** is one of three related chemokines that specifically activate chemokine receptor CCR3. **Chemokine receptor CCR3** may contribute to the eosinophil accumulation in atopic diseases.

5.2. Parallel triple extraction

Our approach aims at distilling the maximum available information from texts that can be used directly for end-user applications. To this end, we integrated three of the most popular OIE systems, each based on a different extraction strategy, into a single module. These complementary OIE engines were chosen both in terms of underlying architecture (i.e. clause-based, deep neural network-based) and also regarding the targeted corpus (i.e. some extractors are focusing on numerical context while others attend to conjunctive sentences). The advantage of having more triples is exploited during the output modification process (Section 7), providing the option to the end-user of having a high recall system, based on the needs of each use case. A brief explanation of the intuition behind each system is provided below:

- *Open IE 5.1* [44] is a successor to the Ollie learning-based information extraction system [15]. It is based on the combination of four different linguistics-based and learning-based OIE tools; namely CALMIE (specializing in triple extraction from conjunctive sentences) [45], RelNoun (for noun relations) [46], BONIE (for numerical sentences) [47], and SRLIE (based on semantic role labeling) [48].
- *ClausIE* [14] follows a clause-based approach, first identifying the clause type of each sentence and then applying specific proposition extraction based on the corresponding grammatical function of the clause's constituents. It also considers nested clauses as independent sentences. Because ClausIE detects useful pieces of information expressed in a sentence before representing them in terms of one or more extractions, it is especially useful in splitting complex sentences into many individual triples.
- *AllenNLP OIE* system [40] formulates the triple extraction problem as a sequence BIO tagging problem and applies a bi-LSTM transducer to produce OIE tuples, which are grouped by each sentence's predicate [49]. Given that it relies on supervised learning and contextualized word embeddings to produce independent probability distributions over possible BIO tags for each word, it has the potential of discovering richer and more complex relations.

We provide an example to showcase the supplementary results of the aforementioned engines which comprise the learning-based extractor in Table 3. The example showcases a triple extraction task on a complex sentence, containing both *independent* and *multiple dependent clauses*. The first column shows the list of derived triples and the second column presents the extraction engine that managed to identify each triple. It is evident that while some of the produced triples remain concise and highly-informative (e.g. first and last extraction), others (second and third) contain redundant information with low marginal utility with respect to the use case. Even longer triples, however, can be useful using a relevant post-processing approach; a feature that is addressed by the Triple Refinement module described in the next section. In general, each engine that comprises the learning-based extractor has diverse quality characteristics from which we can benefit from while using their combined extractions.

Table 3
Parallel triple extraction example from the learning-based extractor, using different engines (O: Open IE, C: ClausIE, A: AllenNLP OIE).

Original sentence: The protein encoded by KIF1A gene is a member of the kinesin family and functions as an anterograde motor protein that transports membranous organelles along axonal microtubules.	
Derived triple	Engine
The protein; be encoded ; by KIF1A gene	O, C, A
The protein encoded by KIF1A gene ; is ; a member of the kinesin family and functions as an anterograde motor protein of the kinesin family that transports membranous organelles along axonal microtubules	C
The protein encoded by KIF1A gene ; is ; a member of the kinesin family and functions as an anterograde motor protein	O, A
An anterograde motor protein ; transports ; membranous organelles along axonal microtubules	O, A

Examples such as this pose some of the most challenging research cases, as it is usually difficult to identify all possible individual clauses using a single extraction approach. However, the complementary strategy of the learning-based extractor manages to capture triples from different parts of the sentence, with a portion of them focusing on the main clause and the rest on the dependent clauses.

6. Triple refinement

Once both triple extraction engines have output a set of triples for a given sentence, they are combined into a single unified set, which aims to maintain the *high precision of the rule-based engine*, but *increases recall* using the triples output from the learning-based engine (which is designed to produce more triples, but with lower precision). To do this, the learning-based triples are first passed through a triple refinement procedure.

This procedure maps the output triples onto the sentence's syntactic dependency tree. For every triple, each node on the tree is marked if it is part of the subject, predicate or object. We design a set of edge-rules, similarly to the rule-based extractor, which signify if an edge is valid between two marked nodes. There are three sets of rules for subject, predicate and object nodes.

For example, in the triple from Table 3:

```
(The protein encoded by KIF1A gene ; is ;
 a member of the kinesin family and
 functions as an anterograde motor protein)
```

The object nodes `member` and `functions` are connected by a *conjunction* relation, which is not a valid object edge, according to the refinement ruleset. The whole branch of the `functions` sub-tree is pruned, and the object becomes: `a member of the`

`kinesin family`. This procedure uses the *inclusion rules* described in Table 1 to determine which branches are kept in the final tree. In this sense, the triple refiner uses the same rules as the Rule-Based Extractor to determine which tokens and phrases are relevant. However, the crucial difference is that the Rule-Based Extractor will completely disregard the entire sub-tree of an ambiguous phrase, while the Learning-Based Extractor plus Triple Refiner will include ambiguous branches, but prune them to remove irrelevant terms. Because of this, while the Rule-Based Extractor may miss ambiguous elements, and the Learning-Based Extractor may include irrelevant elements in its output, the Triple Refiner prunes irrelevant parts of a given triple to create a best-of-both-worlds approach. This procedure is described formally in Algorithm 2, where V represents the table of *inclusion rules* from Table 1.

Algorithm 2 Triple Refiner

Input: T, a set of triples, (S, P, O), for a sentence;
t, a dependency tree for that sentence;
V, a mapping from $\{subject, predicate, object\}$ to a list of valid dependencies
Output: R, a set of refined triples

$R \leftarrow \emptyset$
for all $(S, P, O) \in T$ **do**
  $t' \leftarrow \text{Copy}(t)$
  for all $n \in \text{DepthFirstSearch}(t')$ **do**
    if $n \in S$ **then** $n.mark = subject$
    else if $n \in P$ **then** $n.mark = predicate$
    else if $n \in O$ **then** $n.mark = object$
    else $n.mark = none$
  for all $n \in \text{BreadthFirstSearch}(t')$ **do**
    if $n.mark = n.parent.mark$ **then**
      if $d(n, n.parent) \notin V[n.mark]$ **then**
        $n.mark \leftarrow none$
        for all $n' \in \text{DepthFirstSearch}(n)$ **do**
          $n'.mark \leftarrow none$
  $r \leftarrow \{subject : \emptyset,\ predicate : \emptyset,\ object : \emptyset\}$
  for all $n \in \text{DepthFirstSearch}(t')$ **do**
    $r[n.mark] \leftarrow r[n.mark] \cup \{n\}$
  $R \leftarrow R \cup \{r\}$

Because of this, new triples can be obtained which the rule-based extractor was unable to find, while maintaining the high precision of the rule-based system. Additionally, because the triples are mapped to the dependency tree, they can be annotated with additional information. For example, the object in the previous triple now becomes:

```
a member of the kinesin family
```

Where `kinesin` can be easily isolated as a compound noun from the triple, for later entity linking. Mapping all triples to a dependency tree also allows us to apply the output modification procedures, described in the following section, to all triples, regardless of which engine they originated from.

After the refinement procedure, the triples are merged into a single set. If there are potential duplicates, the system favors the rule-based triple, in order to maintain high precision. For example, if two triples from different extractors have the same predicate for a given sentence, the rule-based one is chosen.

7. Output modification

Once the triple refinement is complete, and each triple is also mapped to a dependency tree, we can modify the final output based on several settings, which activate or disable a small number of auxiliary rules. The core rules and functionality remain generic and do not need to be modified, but these additional optional rules allow for extra flexibility if the text domain requires it.

This allows our system to *adapt to* a wider variety of *text domains and relational databases*. If, for example, longer entities with additional context are required, a switch can be turned on which will provide a *more expressive output*. Or, if the input textual data is highly structured, compound entities can be split into smaller-sub entities for better matching. However, this modification may produce noisy results on less-structured text, since complex or ambiguous conjunctions are difficult to parse accurately, so it can be turned off in these cases. These modifications are described formally in Algorithm 3. We will discuss the details of the algorithm in the following subsections.

Algorithm 3 Output Modification Procedures

function EnhancedPredicates(t)
  for all $n \in \text{BreadthFirstSearch}(t)$ **do**
    if $\{c \mid c \in n.children,\ d(c, n) = dobj\} = \emptyset$ **then**
      for all $c \in n.children$ **do**
        if $d(c, n) = cop$ **then**
          $d(c, n) \leftarrow advmod$; $c \leftarrow n$; $n \leftarrow c$
        if $d(c, n) = nmod$ **and** n is verbal **then**
          $d(c, n) \leftarrow vmod$
  return t
function EntityContext(t)
  $T \leftarrow T \cup \{t\}$
  for all $n \in \text{DepthFirstSearch}(t)$ **do**
    for all $c \in n.children$ **do**
      if $d(c, n) \in \{acl, aclrelcl, advcl\}$ **then**
        $T \leftarrow T \cup \text{Copy}(c)$
        $n.children = n.children \setminus \{c\}$
  return T
function SplitTriples(t)
  $t' \leftarrow \text{Copy}(t)$
  for all $n \in \text{DepthFirstSearch}(t')$ **do**
    $n.children \leftarrow \{c \mid c \in n.children,$
      $d(c, n) \notin \{conj, cc\}\}$
  $T \leftarrow T \cup \{t'\}$
  for all $n \in \text{BreadthFirstSearch}(t)$ **do**
    if n is not verbal **then**
      for all $c \in n.children$ **do**
        if c is nominal **and** $d(c, n) = conj$ **then**
          $n.children \leftarrow \{c' \mid c' \in n.children,$
            $d(c', n) \notin \{conj, cc\}\}$
          $n.children \leftarrow n.children \cup c.children$
          $n.token \leftarrow c.token$
          $t' \leftarrow \text{CopyTree}(t)$; $T \leftarrow T \cup \{t'\}$
  return T

7.1. `enhanced_predicates` Setting

When `enhanced_predicates` is enabled (see first function in Algorithm 3), the predicate contains additional descriptive information, and the subject and object are simplified. For the sentence:

> Blood E2F3 mRNA levels were significantly higher in lung cancer patients when compared to either patients with benign lung diseases or healthy subjects.

The default version keeps the shorter extracted predicate:

```
(Blood E2F3 mRNA levels ;
were ; significantly higher in lung cancer patients)
```

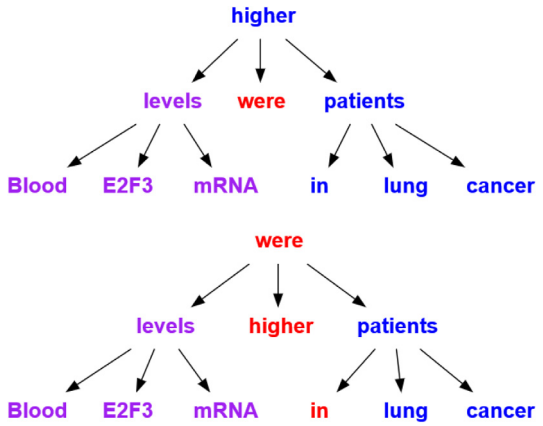


Fig. 4. Section of a dependency tree with and without `enhanced_predicates` enabled (subject, predicate and object are marked in purple, red and blue, respectively). LILLIE enables enhancement of extracted triples with more descriptive information. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

While the `enhanced_predicates` version is as follows:

```
(Blood E2F3 mRNA levels ;
 were significantly higher in ; lung cancer patients)
```

This is illustrated in Fig. 4, where the predicate node (`were`) has been promoted to the root of the tree, and the original root, (`higher`) has been changed to an adverbial modifier. This expands the tokens in the predicate (shown in red).

7.2. *entity_context* Setting

If the textual input data is highly descriptive, and many entities include additional contextual (temporal, location, etc.) information in the subject or object, we can disable the `entity_context` option to exclude potentially noisy additional clauses or phrases (see second function in Algorithm 3). This aids in entity matching, as the relevant tokens can be extracted more efficiently. Without this option disabled, the output triple from the previous example would be:

```
(Blood E2F3 mRNA levels ;
 were significantly higher in ;
 lung cancer patients when compared
 to either patients with benign
 lung diseases or healthy subjects)
```

The benchmark evaluation results in Table 5 have the `entity_context` switch enabled, since human annotators tend towards more contextual labeling. However, we disable this switch when using our system for structured data integration, as the additional information causes excess noise during the entity linking procedure. This ease of adaptability to different domains and contexts, using the output modification switches, is one of the key strengths of the LILLIE system.

7.3. *split_triples* Setting

We can also split or merge conjunctive phrases into sub-triples (see third function in Algorithm 3). This option is best-suited for well-structured textual data, as complex conjunctions can introduce errors (i.e. lower precision) if the text is not well-formed. Consider the following sentence:

> Long non-coding RNA CCAT2 plays an important role in tumorigenesis, tumor growth and metastasis.

We can extract three separated triples for each sub-entity:

```
(Long non-coding RNA LncRNA CCAT2 ; plays ;
 an important role in tumor growth)
(Long non-coding RNA LncRNA CCAT2 ; plays ;
 an important role in metastasis)
(Long non-coding RNA LncRNA CCAT2 ; plays ;
 an important role in tumorigenesis)
```

Or, merge them into a single entity:

```
(Long non-coding RNA LncRNA CCAT2 ; plays ;
 an important role in tumorigenesis tumor growth and metastasis)
```

We demonstrate the effect of the `split_triples` enhancement in Table 7. When this switch is enabled, we gain a higher number of triples that can be linked to entities for database insertion. In this case, the enhancement was effective due to the linguistically precise nature of the scientific texts used.

It should be noted that, while the combined effect of the proposed *triple refinement* and *output modification* processes can be considered similar to that of knowledge base canonicalization (i.e. the problem of mapping each entity to its canonical form to reduce ambiguity or redundancy) [50], our approach significantly differs from the established line of work [51,52] as it allows *full control of the auxiliary information captured by the extractors, based on user preference*.

8. Entity linking and database integration

The final step of our end-to-end system is to integrate the extracted triples into a relational database, in order to increase the contextual value of our information extraction pipeline. Even though our entity linking and database integration approach is generic, we describe a specific use case of applying our approach in the domain of medical research. We have chosen this use case to demonstrate LILLIE's impact of solving a challenging real-world problem [8] at the intersection of academia and industry based on solid theoretical foundations.

We followed an *entity linking strategy* to correlate triples subjects and objects with a specific anatomical entity from the Uberon [53] cross-species anatomy ontology (e.g. pharynx = `UBERON:0006562`) and with a specific biomarker from the OncoMX [54] cancer mutation and expression knowledgebase (e.g. Keratin 8 = `KRT8`). Finally, we *integrated the linked triples into the relational database* called OncoMX.

8.1. Entity linking

Since the output of our triple extractor and refiner contains additional annotations, such as compound or adjectival components, we can efficiently link textual entities from our extracted triples with existing entities in a database. For example, with the entity `long non-coding RNA CCAT2`, we can separate this into four sub-entities:

```
long non-coding RNA CCAT2
RNA CCAT2
CCAT2
long non-coding CCAT2
```

Each of these new entities has a different degree of granularity, and we can now match on the most specific entity available. For example, if *long non-coding CCAT2* was present in our database, we select this match; however, if only the less specific *CCAT2*

	id	label
0	UBERON:0002539	pharyngeal arch, pharyngeal arches, branchial ...
1	UBERON:0002544	digit, acropodial unit, digit (phalangeal port...
2	UBERON:0003221	phalanx, digit long bone, long bone of digit, ...
3	UBERON:0002228	rib, dorsal rib, pleural rib, ribs
4	UBERON:0002412	vertebra, vertebra bone, vertebrae

Fig. 5. Sample of Uberon anatomical entities, showing ids and labels (i.e. exact names and synonyms) of humans and animals.

entity was present, we can still find a match. This approach has several advantages: it allows for a more efficient search, and can handle *n*-grams split over several sub-phrases. It also does not require a similarity measure, which enhances precision, while maintaining high recall.

Using the Uberon ontology, we then concatenated two properties of the ontology (`label`, `hasExactSynonym`) to create a simple dataframe of the following structure as shown in Fig. 5. The column `label` contains a list of different names (official label and synonyms) for each anatomical entity and was used to match the extracted triples with Uberon entities.

Similarly, we collected the gene names from the OncoMX database comprising a table of 809 gene records. We then ran an iterative process to match one or more mentions of gene names with each triple's subject or object.

8.2. Database integration and enrichment

We applied the aforementioned entity linking approaches on the extracted triples in order to enrich the existing OncoMX database with the new information stemming from our information extraction system. In particular, we could increase the information context of the OncoMX database via linking it with literature mentions of genes that are affecting cancer development in specific Uberon anatomical entities, as shown in Fig. 6. Each row contains a Pubmed ID (*pmid*) that corresponds to a medical article, a gene name (*gene*), an Uberon entity ID (*uberon*) and name (*uberonname*), as well as the extracted relational triple in *subject–predicate–object* format. The enriched database can then be used for querying structured and previously unstructured data via a single common query interface such as SQL or in natural language.

9. Experiments

To evaluate our system, we first measure the performance of our triple extractor against two state-of-the-art systems, OpenIE6 [55] and IMoJIE [56], on two standard benchmark data sets. Next, we use the PubMed abstracts dataset to demonstrate the qualitative advantages of our enhancements, in comparison to these systems and to show that our *approach generalizes well for a diverse set of datasets*. Lastly, we show how our data can be successfully queried in a relational setting from a database enriched with triples. For reproducibility of our results, we make the source code of LILLIE available.[3]

9.1. Datasets

The performance of a triple extraction system is assessed as follows: for a given sentence, a system's extracted triples are compared with a set of gold-standard triples, selected by human annotators, and precision and recall are measured on term-level. We use two datasets of annotated sentences for our experiments:

Table 4
Sample extractions on the domain-generic CaRB benchmark dataset.

Original sentence
Warner Communications Inc., which is being acquired by Time Warner, has filed a $1 billion breach-of-contract suit against Sony.
Generated triples
(1) Warner Communications Inc. ; is being acquired by ; Time Warner (2) Warner Communications Inc. ; has filed ; a $ 1 billion breach of contract suit against Sony
Original sentence
The fuselage had an oval cross-section and housed a water-cooled inverted-V V-12 engine.
Generated triples
(1) The fuselage ; housed ; a water-cooled inverted-V V-12 engine (2) The fuselage ; had ; an oval cross-section
Original Sentence
Although Heathrow authorities have been watching a group of allegedly crooked baggage handlers, the Gauguin may be 'lost'.
Generated triples
(1) Heathrow authorities ; have been watching ; a group of allegedly crooked baggage handlers (2) the Gauguin ; may be ; 'lost'

- The *CaRB dataset* [36] contains 1282 open-domain sentences, divided into two sets of 641 sentences, for development and testing, respectively.
- *Re-OIE16* [37] is an updated subset of OpenIE16 [57], an earlier corpus, with all sentences re-annotated to better reflect the needs of OIE. Similar to CaRB, this contains 600 open-domain sentences.

A sample of the sentences contained in CaRB, and the triples we extract, are shown in Table 4.

For further demonstrating the generalizability of our approach, we also perform extractions on a set of medical journal papers, taken from a variety of disciplines. The *PubMed abstracts* dataset contains 116,049 sentences across 38,703 abstracts and paper titles.

9.2. Performance of LILLIE's triple extraction pipeline

We first evaluate the triple extraction portion of our system LILLIE using the CaRB Evaluator[4] on the CaRB test set. In addition, we adapt the Re-OIE16 annotations to be used with the CaRB evaluator, in order to provide consistent results.

Initially, we used the CaRB development set, comprising 50% of the sentences, for development and tuning of the rule-based extractor and refiner. We developed a set of generic, domain-independent linguistic rules by analyzing a randomly-chosen sample of 30 sentences for linguistic patterns, then tested these rules on the remainder of the CaRB development set, to ensure maximum generalizability. These rules remained the same throughout all further evaluations and experiments. We then evaluated the performance of our system against the current state-of-the art systems on the testing portion of the CaRB set, and the *previously unseen ReOIE16 benchmark*. Since these two datasets are open-domain, and were not used for training or development, we believe our results on these benchmarks show that our system, and its components, such as the rule sets, are generic and adaptable to many different domains.

The results are shown in Table 5. We observe the most *significant improvements for our approach (LILLIE)* in the AUC scores, with an approx. 6% increase *over both state of the art systems*

[3] Source code of LILLIE: https://github.com/OIELILLIE/LILLIE.

[4] https://github.com/dair-iitd/CaRB.

pmid integer	gene text	uberon text	uberonname text	subject text	predicate text	object text
24046070	IDH1	UBERON:0002048	lung	idh1	can be used as	a plasma biomarker for the diagnos...
20421816	EGFR	UBERON:0000021	cutaneous appe...	non-small c...	is sensitive to	the small molecule egfr tyrosine kin...
23719586	KRAS	UBERON:0000006	islet of Langerh...	pdx1-driven ...	could induce	islets of langerhans defects
25700797	EGFR	UBERON:0014899	anterolateral lig...	epidermal gr...	are present in	10-20 % of all non-small-cell lung c...
21271382	MGMT	UBERON:0001155	colon	the methylat...	were 78.3 52.5...	colorectal cancer in colon adenomas

Fig. 6. Enriched OncoMX database after information extraction, entity linking and database integration.

Table 5
Evaluation results for triple extraction on the CaRB test set and Re-OIE16, compared to OpenIE6 and IMoJIE. LILLIE (our approach) substantially outperforms state-of-the-art systems. AUC = Area under Curve, P = Precision, R = Recall.

	CaRB				Re-OIE16			
	AUC	P	R	F1	AUC	P	R	F1
LILLIE	**.391**	.604	**.487**	**.539**	**.543**	**.685**	**.645**	**.664**
IMoJIE	.333	**.647**	.456	.535	.483	.653	.584	.617
OpenIE6	.337	.589	.477	.527	.523	.642	.612	.627

OpenIE6 and IMoJIE in the CaRB dataset, and an increase across all metrics in the Re-OIE16 dataset. The precision/recall balance achieved by the triple refiner and combiner of LILLIE is reflected in the improved F1 scores on both datasets. In particular, precision and recall are well-matched on Re-OIE16, despite the system being tuned on the CaRB development set, and ReOIE16 was not used during the training or development of our system, which demonstrates good generalizability.

This result is noteworthy as tuning the output of the triple extraction component for these benchmarks was not an isolated goal, with the component instead tailored to be part of our bigger end-to-end system. The generalizibility of our domain-independent information extraction approach is also qualitatively depicted in Table 4 via a diverse selection of examples taken from the CaRB test set.

9.3. Ablation study

In Table 6, we show the results of each component of our system, in accordance with the architecture shown in Fig. 2. We give AUC and F1 scores on both CaRB and ReOIE testing sets, in order to show the effect of each individual component.

The individual scores for the Rule-Based and Learning-Based components are shown, along with the effects of pre-processing and co-reference resolution, respectively. The Rule-Based pre-processing *increases the F1 and AUC scores* on both datasets, and the in-place co-reference resolution *increases the F1 scores* on both data sets, while somewhat *degrading AUC*. This is due to the fact that the CaRB and ReOIE datasets contain single sentences, rather than longer texts, so the positive effects of co-reference resolution are less apparent. Nevertheless, the addition of this component improves the overall results when the triple refiner is applied.

Finally, we analyze the performance of combining the Rule-Based (RB) and Learning-Based (LB) approaches as indicated in Table 6 by "Combination of RB and LB". In particular, we demonstrate the improvements made by the Triple Refinement process by showing the results of a simple union of the triples from both components. A raw combination of the high-recall triples and high-precision triples yields a degradation of both F1 and AUC scores on both sets. However, when *using the triple refiner, all metrics are improved*, with AUC showing a significant increase on both sets. These results demonstrate that *each individual component of our system provides a net-positive improvement* in benchmark scores, and, when combined together, result in a substantial improvement overall.

Table 6
Ablation study of LILLIE. AUC and F1 scores for each individual component of our system.

	CaRB		ReOIE	
	AUC	F1	AUC	F1
Rule-Based (RB)				
–without pre-processing	.337	.503	.459	.624
–with pre-processing	.370	.531	.507	.657
Learning-Based (LB)				
–without coref resolution	.381	.442	.470	.440
–with coref resolution	.358	.457	.457	.500
Combination of RB and LB				
–raw combination	.365	.424	.478	.448
–refined combination	**.391**	**.539**	**.543**	**.664**

9.4. Error analysis

In general, LILLIE encounters errors in three main areas: firstly, in *qualified phrases*, such as "*Research shows that...*"; secondly, in sentences which are *grammatically ambiguous* to parse; thirdly, in triples *requiring inference*. We will discuss these error areas in more detail below.

(1) Qualified phrases: A sentence containing a qualified phrase is one in which the main clause is not necessarily implied to be factual. In the sentence "*This protein leads to tumor growth.*", the meaning is unambiguous. However, if the sentence were "*Some studies claim that this protein leads to tumor growth.*", the sub-clause is not necessarily implied as fact.

Consider the following sentence from PubMed as an example of a sentence with a qualified phrase:

> The prevailing view of CD73 is that it is overexpressed in tumors.

LILLIE extracts the following triple:

```
(CD73 ; is ; overexpressed in tumors)
```

This is not necessarily a definitively true statement, since it is qualified as an opinion, and outputting this triple may result in ambiguous information being added to a database. Our engine does not account for such qualifying phrases, and outputs these triples as facts. However, the outputs of OpenIE6 and IMoJIE will produce similar unqualified triples, and currently this remains an open problem in Information Extraction.

(2) Grammatically ambiguous sentences: Our approach encounters errors in cases where the sentence is grammatically complex, such as the following, taken from the CaRB development set:

> US 258 and NC 122 parallel the river north before the two routes diverge northeast of Tarboro.

Here, the dependency parser may fail to produce the necessary parse — in this case, due to the term "*parallel*" being misinterpreted as a noun, rather than a verb. However, in use cases such

SQL Query

```
SELECT distinct gene, uberonname, uberon
FROM triples_fully_linked_v2
WHERE
(predicate like '%overexpress%' or
(predicate like '%express%' and (subject like '%over%'  or
     triples_fully_linked_v2.object like '%over%')) or
     (subject like '%overexpress%' or
     triples_fully_linked_v2.object like '%overexpress%'))
and (subject like '%cancer%' or
     triples_fully_linked_v2.object like '%cancer%') and
polarity='TRUE' and uberonname='breast'
```

Results

	gene text	uberonname text	uberon text
1	EGFR	breast	UBERON:0000310
2	EPCAM	breast	UBERON:0000310
3	CD24	breast	UBERON:0000310
4	NR4A1	breast	UBERON:0000310
5	ECM1	breast	UBERON:0000310
6	RAB5A	breast	UBERON:0000310
7	CXCL13	breast	UBERON:0000310
8	UBQLN1	breast	UBERON:0000310
9	MUC1	breast	UBERON:0000310
10	RET	breast	UBERON:0000310
11	RHOA	breast	UBERON:0000310
12	ERBB2	breast	UBERON:0000310
13	CCND1	breast	UBERON:0000310

Fig. 7. Example of a SQL query on the enriched OncoMX database. LILLIE enables querying structured and (previously) unstructured data from a single common query interface.

as the PubMed abstracts, where text is often unambiguous, this issue occurs rarely.

(3) Triples requiring inference: For cases where inference is required, our extractor has a lower recall than other engines. Because of our aim to build a high-precision engine for database integration, we make no attempt to infer triples that are not directly implied by the text, as this was found to add additional noise and degrade precision. For example, consider the following title of a paper on PubMed:

> Association of Leptin, Visfatin, and Adiponectin With Renal Cell Carcinoma

There may be indirect triples, such as:

```
(Leptin ; is associated with ; Renal Cell Carcinoma)
```

Other systems attempt to extract such triples; however, when testing additional inference methods, we found them to degrade precision, and introduce many noisy superfluous triples. As such, we were unable to maintain our precision–recall balance in these cases. This is an area for further study, as a high-precision method of extracting such information would be valuable for entity linking.

9.5. Positive effect of triple enhancements

Using examples from the PubMed abstracts dataset, we now demonstrate the effects of our various triple enhancement procedures, in comparison to the triples output from the OpenIE6 and IMoJIE systems. To begin with, we show the output of both existing systems on our main example sentence:

> Long non-coding RNA CCAT2 plays an important role in tumorigenesis, tumor growth and metastasis.

The IMoJIE system, while achieving high precision on the test sets, does not output split triples of the form demonstrated in Section 7. Instead, it groups all conjunctive entities into a single triple:

```
(Long non-coding RNA ; plays ;
an important role in tumorigenesis , tumor growth and metastasis)
```

This form of triple reduces precision during the entity linking process — particularly in cases where the input text is sufficiently well-formed to reliably split triples. On the other hand, the OpenIE6 system produces a similar output to our method:

```
(Long non-coding RNA ; plays ; an important role in metastasis)
```

However, our `split_triples` flag allows this behavior to be enabled or disabled depending on the nature of the input data, giving a balance between extraction precision and entity linking precision. Additionally, the pre-processing steps described in Section 4 allow for verbal conjunctions, such as in:

> 2-O-Methylmagnolol Upregulates the Long Non-Coding RNA, GAS5, and Enhances Apoptosis in Skin Cancer Cells

Where the IMoJIE and OpenIE6 systems output only one triple for this sentence:

```
(2-O-Methylmagnolol ; Upregulates ;
 the Long Non-Coding RNA , GAS5 ,
 and Enhances Apoptosis in Skin Cancer Cells)
```

SQL Query

```
SELECT distinct gene, uberonname, uberon
FROM triples_fully_linked_v2
WHERE
(predicate like '%overexpress%' or
(predicate like '%express%' and (subject like '%over%' or
    triples_fully_linked_v2.object like '%over%')) or
(subject like '%overexpress%' or
    triples_fully_linked_v2.object like '%overexpress%')) and
(subject like '%cancer%' or triples_fully_linked_v2.object like '%cancer%')
        and polarity='TRUE'
LIMIT 20
```

Results

	gene text	uberonname text	uberon text
1	AMACR	prostate gland	UBERON:0002367
2	NR4A1	breast	UBERON:0000310
3	MDK	lung	UBERON:0002048
4	HMGB1	lung	UBERON:0002048
5	LAPTM...	lung	UBERON:0002048
6	ERBB2	cutaneous appendage	UBERON:0000021
7	RHOA	breast	UBERON:0000310
8	GPI	prostate gland	UBERON:0002367
9	IGF2	tissue	UBERON:0000479
10	AGR2	prostate gland	UBERON:0002367
11	ECM1	tissue	UBERON:0000479
12	EGFR	lung	UBERON:0002048
13	EPCAM	prostate gland	UBERON:0002367
14	SLPI	tissue	UBERON:0000479
15	KLK6	tissue	UBERON:0000479
16	EGFR	breast	UBERON:0000310
17	EGFR	colon	UBERON:0001155
18	CD24	breast	UBERON:0000310
19	HSP90...	lung	UBERON:0002048
20	CD59	lung	UBERON:0002048

Fig. 8. Example of a SQL query on the enriched OncoMX database. It corresponds to the equivalent natural language question: "Find all anatomical entities where genes are over-expressed due to some cancer reported in the literature".

Our system uses the subject duplication procedure shown in Fig. 3 and Algorithm 1 to obtain two triples, one for each object entity:

```
(2-O-Methylmagnolol ; Upregulates ; the Long Non-Coding RNA GAS5)
(2-O-Methylmagnolol ; Enhances ; Apoptosis in Skin Cancer Cells)
```

10. Database enrichment and querying

With these high-precision triples, we can more accurately link textual mentions of named entities to their corresponding entries in a knowledge base. This allows us to perform new queries that were not supported by the original database. We showcase this capability by leveraging the raw information contained in PubMed abstracts to enrich the OncoMX database from Section 8. Using the OncoMX database, we can execute structured relational queries on unstructured, textual information stored in medical articles. Such an example is provided in Fig. 7, which corresponds to the equivalent natural language question: "What are the genes over-expressed in breast cancer that are reported in the literature?"

We are able to answer this query by exploiting the triples extracted from the PubMed medical articles (unstructured data) and their mapping to genes and anatomical entities (structured data). The goal of our query is to find all literature cases that include "over-expression" of a gene, specifically on breast cancer. The result is a subset of 13 genes that are reported as being over-expressed in breast cancer cases.

In a similar manner, we can extend our search to find all anatomical entities where genes are over-expressed due to cancer according to the literature. The results of this query are shown in Fig. 8. The number of returned rows is limited to 20 from the original 70 for visualization purposes.

Finally, we can focus our search on finding all literature cases derived from the triple extraction of Pubmed articles which include the keywords "cancer" and "biomarker" in the extracted triples. The results shown in Fig. 9 contain the captured genes and anatomical entities, along with their corresponding subject–predicate–object relational triples.

In order to quantify the effect of our tight integration between triple extraction and entity linking, we also perform a comparison between each system on this same dataset. For this experiment, we run OpenIE6, IMoJIE and two versions of LILLIE (with `split_triples` enabled and disabled, respectively) on the set of PubMed abstracts, and perform the entity linking procedure described in Section 8. For OpenIE6 and IMoJIE, we use an n-gram based search over the Uberon and OncoMX databases, as they do not provide the annotated triples of our system. This procedure simply searches the databases with all possible n-grams from the subject and object, until the longest match is found, rather than deriving the n-grams from the syntactic structure of the triple.

The dataset consists of 38,703 abstracts, comprising 116,049 sentences. We extract triples from each sentence, and attempt to link each triple with the structured database. Linked triples are ones that contain both an Uberon anatomical entity and an

SQL Query

```
SELECT distinct gene, uberonname, uberon
FROM triples_fully_linked_v2
WHERE
(predicate like '%overexpress%' or
(predicate like '%express%' and (subject like '%over%'  or
     triples_fully_linked_v2.object like '%over%')) or
     (subject like '%overexpress%' or
     triples_fully_linked_v2.object like '%overexpress%'))
and (subject like '%cancer%' or
     triples_fully_linked_v2.object like '%cancer%') and
polarity='TRUE' and uberonname='breast'
```

Results

gene text	uberon text	uberonname text	subject text	predicate text	object text	polarity text
TNF	UBERON:0013756	venous blood	stress variables profile of antioxida...	were biochemically assessed from	venous blood of fifty ovarian c...	TRUE
TNF	UBERON:0000310	breast	tnf polymorphisms	could serve as	useful predictive biomarkers f...	TRUE
CRYAB	UBERON:0002367	prostate gland	lower cryab expression	is a prognostic biomarker for	several types of cancer such ...	TRUE
FABP5	UBERON:0000029	lymph node	fatty acid-binding protein 5 fabp5	was found in previous study to bi...	be a potential for lymph node ...	TRUE
FABP5	UBERON:0002391	lymph	fatty acid-binding protein 5 fabp5	was found in previous study to bi...	be a potential for lymph node ...	TRUE
KLK3	UBERON:0001628	posterior commu...	klk3 gene products like human pro...	are important biomarkers in	the clinical diagnosis of prost...	TRUE
TNF	UBERON:0000178	blood	stress variables profile of inflamma...	were biochemically assessed from	venous blood of fifty ovarian c...	TRUE
CRYAB	UBERON:0002367	prostate gland	lower cryab expression	is a prognostic biomarker for	several types of cancer such ...	TRUE
S100A4	UBERON:0002107	liver	nuclear expression of the calcium-...	is a biomarker of	increased invasiveness in cho...	TRUE
XRCC1	UBERON:0002048	lung	xrcc1 genetic polymorphism	acts	a potential biomarker for lung ...	TRUE
CCND1	UBERON:0000310	breast	ccnd1 mutations	may serve as	biomarkers for early diagnosi...	TRUE
CCND1	UBERON:0000310	breast	ccnd1 and cdk4 mutations	may serve as	biomarkers for early detection...	TRUE
AMACR	UBERON:0002367	prostate gland	expression of the alpha-methylacyl...	has been established as	a specific biomarker for the di...	TRUE
CRYAB	UBERON:0000974	neck	lower cryab expression	is a prognostic biomarker for	several types of cancer such ...	TRUE
QSOX1	UBERON:0000310	breast	qsox1	could be posited as	a new biomarker of good prog...	TRUE
EGFR	UBERON:0000310	breast	egfr expression levels	are	key biomarkers for breast can...	TRUE
MMP1	UBERON:0002048	lung	the mmp1 -1607 1g allele	is a non-significant protective bio...	lung cancer in taiwan	TRUE
TNF	UBERON:0000178	blood	stress variables profile of antioxida...	were biochemically assessed from	venous blood of fifty ovarian c...	TRUE
CRYAB	UBERON:0000974	neck	lower cryab expression	is a prognostic biomarker for	several types of cancer such ...	TRUE
CCND1	UBERON:0000310	breast	ccnd1 and cdk4 mutations	may serve as	biomarkers for early diagnosi...	TRUE

Fig. 9. Example of a SQL query on the enriched OncoMX database. It allows us to search for cancer biomarkers based on specific keywords found in the literature, showcasing LILLIE's information extraction capabilities.

Table 7
The number of extracted triples on the PubMed abstracts dataset for each system, and the number of these triples that link with both an Uberon anatomical entity and an OncoMX gene symbol. LILLIE shows a higher ratio of relevant triples (linked triples divided by extracted triples) than the state-of-the-art systems.

	Extracted triples	Linked triples
LILLIE	206,096	3513
LILLIE (`split_triples` = 0)	117,290	2448
OpenIE6	247,072	3110

Table 8
The number of extracted triples on a sample of 1000 PubMed abstracts, and the number of these triples that link with both an Uberon anatomical entity and an OncoMX gene symbol.

	Extracted triples	Linked triples	Time per sentence
LILLIE	5648	71	7.54
OpenIE6	6675	50	1.46
IMoJIE	4565	49	14.19

OncoMX gene symbol, with one in the subject and the other in the object, irrespectively. Partial matches are not recorded. The results of this are shown in Table 7 for OpenIE6 and LILLIE. However, we were unable to run the full dataset with IMoJIE, which encountered memory issues on the large amount of data, so we show the results for a sample of 1000 abstracts (3035 sentences) in Table 8. We also show the average speed of each system, tested on an Intel Core i7-7700HQ 2.80 GHz CPU, with 32 GB RAM and NVIDIA GeForce GTX 1050 GPU.

The results show that, when using our entity linking procedure, LILLIE achieves a higher ratio of relevant triples in comparison to OpenIE6. In particular, the higher-precision version, with `split_triples` disabled, achieves a comparable amount of linked triples, on less than half the extractions overall. This shows that *our system can additionally be adapted to provide a higher-precision variant*, if required.

We report a slower runtime for our system compared to OpenIE6, and a faster runtime that IMoJIE. However, as shown in Tables 7 and 5, we extract more accurate (higher F1 and AUC scores) and more usable (higher ratio of linked triples) triples overall.

In the future we will explore more advanced entity matching techniques based on transformer neural network architectures such as the ones presented in [33,58]. However, for our end-to-end data processing pipeline, the current entity linking approach showed already promising results.

11. Conclusions

In this paper, we presented LILLIE — an end-to-end system for the enrichment of relational databases with extracted information from unstructured text. We developed a (1) precision-oriented, *linguistics-based*, triple extraction approach using domain-independent generic rules. We combined this approach with a (2) recall-oriented, *learning-based*, triple extraction approach to counter the loss of structural and semantic information. LILLIE not only allows effectively combining these two approaches but also enables *enhancements of the extracted information* via parameterizable postprocessing. Hence, our system is able to adapt to a diverse set of textual domains. Finally, we also leverage entity linking methods to integrate textual entities from our extracted triples into a relational database, thus increasing the contextual value of the extracted entities.

We compared LILLIE's performance with the two popular state of the art OIE systems, IMoJIE and OpenIE6, on the two widely-used benchmark datasets CaRB and Re-OIE16. LILLIE shows a substantial performance gain over the existing systems in terms of AUC, Precision, Recall and F1-score. Moreover, we demonstrated the effects of our triple enhancement processes on a corpus comprising biomedical documents (PubMed abstracts) to highlight the generalizability of our approach.

Future work could investigate the integration of additional learning-based OIE extractors, employing transfer or few-shot learning techniques to enhance and extend information extraction for domain-specific or even multi-lingual corpora for which no training data is available. Another interesting line of work could address entity disambiguation using popular knowledge bases (e.g. DBpedia) or word embedding techniques to resolve complex one-to-many property mappings, further increasing the effectiveness of our system.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This project has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No 863410. The research work of D.P. was supported by the Hellenic Foundation for Research and Innovation (HFRI), Greece under the HFRI PhD Fellowship grant (Fellowship Number: 50, 2nd call).

References

[1] P. Simon, Too Big to Ignore: The Business Case for Big Data, Vol. 72, John Wiley & Sons, 2013.
[2] A. Rogers, The 80% blind spot: Are you ignoring unstructured organizational data? Forbes (2019).
[3] K. Affolter, K. Stockinger, A. Bernstein, A comparative survey of recent natural language interfaces for databases, VLDB J. 28 (5) (2019) 793–819.
[4] U. Brunner, K. Stockinger, ValueNet: A natural language-to-SQL system that learns from database information, in: International Conference on Data Engineering (ICDE), 2021.
[5] A. Hogan, E. Blomqvist, M. Cochez, C. d'Amato, G.D. Melo, C. Gutierrez, S. Kirrane, J.E.L. Gayo, R. Navigli, S. Neumaier, et al., Knowledge graphs, ACM Comput. Surv. 54 (4) (2021) 1–37.
[6] I. Mansuri, S. Sarawagi, Integrating unstructured data into relational databases, in: 22nd International Conference on Data Engineering (ICDE'06), 2006, p. 29, http://dx.doi.org/10.1109/ICDE.2006.83.
[7] A. Jain, A. Doan, L. Gravano, SQL queries over unstructured text databases, in: 2007 IEEE 23rd International Conference on Data Engineering, 2007, pp. 1255–1257, http://dx.doi.org/10.1109/ICDE.2007.368986.
[8] S. Amer-Yahia, G. Koutrika, F. Bastian, T. Belmpas, M. Braschler, U. Brunner, D. Calvanese, M. Fabricius, O. Gkini, C. Kosten, D. Lanti, A. Litke, H. Lücke-Tieke, F.A. Massucci, T.M. de Farias, A. Mosca, F. Multari, N. Papadakis, D. Papadopoulos, Y. Patil, A. Personnaz, G. Rull, A. Sima, E. Smith, D. Skoutas, S. Subramanian, G. Xiao, K. Stockinger, INODE: Building an end-to-end data exploration system in practice [extended vision], 2021, arXiv:2104.04194.
[9] D. Jurafsky, J.H. Martin, Speech and language processing: An introduction to natural language processing, in: Speech and Language Processing: An Introduction to Natural Language Processing Computational Linguistics and Speech Recognition, 2001.
[10] G. Liu, X. Li, J. Wang, M. Sun, P. Li, Extracting knowledge from web text with Monte Carlo tree search, in: The Web Conference 2020 - Proceedings of the World Wide Web Conference, WWW 2020, 2020, http://dx.doi.org/10.1145/3366423.3380010.
[11] M. Banko, M.J. Cafarella, S. Soderland, M. Broadhead, O. Etzioni, Open information extraction from the web, in: IJCAI International Joint Conference on Artificial Intelligence, 2007.
[12] F. Wu, D.S. Weld, Open information extraction using wikipedia, in: ACL 2010 - 48th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference, 2010.
[13] A. Fader, S. Soderland, O. Etzioni, Identifying relations for open information extraction, in: EMNLP 2011 - Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference, 2011.
[14] L. Del Corro, R. Gemulla, ClausIE: Clause-based open information extraction, in: WWW 2013 - Proceedings of the 22nd International Conference on World Wide Web, 2013.
[15] Mausam, M. Schmitz, R. Bart, S. Soderland, O. Etzioni, Open language learning for information extraction, in: EMNLP-CoNLL 2012 - 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, Proceedings of the Conference, 2012.
[16] M. Yahya, S.E. Whang, R. Gupta, A. Halevy, ReNoun: Fact extraction for nominal attributes, in: EMNLP 2014 - 2014 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference, 2014, http://dx.doi.org/10.3115/v1/d14-1038.
[17] K. Kolluru, S. Aggarwal, V. Rathore, Mausam, S. Chakrabarti, IMoJIE: Iterative memory-based joint open information extraction, 2020, http://dx.doi.org/10.18653/v1/2020.acl-main.521, arXiv:2005.08178.
[18] Y. Ro, Y. Lee, P. Kang, Multi^2OIE: Multilingual open information extraction based on multi-head attention with BERT, in: Findings of the Association for Computational Linguistics: EMNLP 2020, Association for Computational Linguistics, Online, 2020, pp. 1107–1117, http://dx.doi.org/10.18653/v1/2020.findings-emnlp.99, URL https://www.aclweb.org/anthology/2020.findings-emnlp.99.
[19] C. Niklaus, M. Cetto, A. Freitas, S. Handschuh, A survey on open information extraction, in: Proceedings of the 27th International Conference on Computational Linguistics, Association for Computational Linguistics, Santa Fe, New Mexico, USA, 2018, pp. 3866–3878, URL https://www.aclweb.org/anthology/C18-1326.
[20] N. Kertkeidkachorn, R. Ichise, T2KG: An end-to-end system for creating knowledge graph from unstructured text, in: AAAI Workshop - Technical Report, 2017.
[21] D. Papadopoulos, N. Papadakis, A. Litke, A methodology for open information extraction and representation from large scientific corpora: The CORD-19 data exploration use case, Appl. Sci. (Switzerland) (2020) http://dx.doi.org/10.3390/app10165630.
[22] D. Papadopoulos, N. Papadakis, N. Matsatsinis, PENELOPIE: Enabling open information extraction for the greek language through machine translation, in: Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Student Research Workshop, Association for Computational Linguistics, Online, 2021, pp. 23–29, URL https://aclanthology.org/2021.eacl-srw.4.
[23] D. Vrandečić, M. Krötzsch, Wikidata, Commun. ACM (2014) http://dx.doi.org/10.1145/2629489.
[24] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, Z. Ives, DBpedia: A nucleus for a web of open data, in: Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2007, http://dx.doi.org/10.1007/978-3-540-76298-0_52.
[25] T.P. Tanon, G. Weikum, F. Suchanek, Yago 4: A reason-able knowledge base, in: European Semantic Web Conference, Springer, 2020, pp. 583–596.
[26] J.R. Finkel, T. Grenager, C. Manning, Incorporating non-local information into information extraction systems by Gibbs sampling, in: ACL-05 - 43rd Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference, 2005, http://dx.doi.org/10.3115/1219840.1219885.
[27] M. Francis-Landau, G. Durrett, D. Klein, Capturing semantic similarity for entity linking with convolutional neural networks, in: 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL HLT 2016 - Proceedings of the Conference, 2016, http://dx.doi.org/10.18653/v1/n16-1150, arXiv:1604.00734.
[28] Y. Sun, L. Lin, D. Tang, N. Yang, Z. Ji, X. Wang, Modeling mention, context and entity with neural networks for entity disambiguation, in: IJCAI International Joint Conference on Artificial Intelligence, 2015.
[29] N. Gupta, S. Singh, D. Roth, Entity linking via joint encoding of types, descriptions, and context, in: EMNLP 2017 - Conference on Empirical Methods in Natural Language Processing, Proceedings, 2017, http://dx.doi.org/10.18653/v1/d17-1284.
[30] P.H. Martins, Z. Marinho, A.F. Martins, Joint learning of named entity recognition and entity linking, in: ACL 2019 - 57th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Student Research Workshop, 2019, http://dx.doi.org/10.18653/v1/p19-2026, arXiv:1907.08243.
[31] L. Logeswaran, M.W. Chang, K. Lee, K. Toutanova, J. Devlin, H. Lee, Zero-shot entity linking by reading entity descriptions, in: ACL 2019 - 57th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference, 2020, http://dx.doi.org/10.18653/v1/p19-1335, arXiv:1906.07348.
[32] L. Wu, F. Petroni, M. Josifoski, S. Riedel, L. Zettlemoyer, Zero-shot entity linking with dense entity retrieval, in: EMNLP 2020 - Conference on Empirical Methods in Natural Language Processing, Proceedings, 2019.

[33] U. Brunner, K. Stockinger, Entity matching with transformer architectures-a step forward in data integration, in: International Conference on Extending Database Technology, Copenhagen, 30 March-2 April 2020, 2020.
[34] Y. Li, J. Li, Y. Suhara, A. Doan, W.-C. Tan, Deep entity matching with pre-trained language models, in: International Conference on Very Large Data Bases, 2021.
[35] Y. Eslahi, A. Bhardwaj, P. Rosso, K. Stockinger, P. Cudré-Mauroux, Annotating web tables through knowledge bases: A context-based approach, in: 2020 7th Swiss Conference on Data Science (SDS), IEEE, 2020, pp. 29–34.
[36] S. Bhardwaj, S. Aggarwal, M. Mausam, CaRB: A crowdsourced benchmark for open IE, in: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), Association for Computational Linguistics, Hong Kong, China, 2019, pp. 6262–6267, http://dx.doi.org/10.18653/v1/D19-1651, URL https://www.aclweb.org/anthology/D19-1651.
[37] J. Zhan, H. Zhao, Span model for open information extraction on accurate corpus, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 34, 2020, pp. 9523–9530.
[38] D. Chen, C.D. Manning, A fast and accurate dependency parser using neural networks, in: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2014, pp. 740–750.
[39] M.-C. De Marneffe, T. Dozat, N. Silveira, K. Haverinen, F. Ginter, J. Nivre, C.D. Manning, Universal stanford dependencies: A cross-linguistic typology, in: LREC, Vol. 14, 2014, pp. 4585–4592.
[40] Allen Institute for AI, AllenNLP coreference resolution demo. URL https://demo.allennlp.org/coreference-resolution.
[41] K. Lee, L. He, M. Lewis, L. Zettlemoyer, End-to-end neural coreference resolution, in: EMNLP 2017 - Conference on Empirical Methods in Natural Language Processing, Proceedings, 2017, http://dx.doi.org/10.18653/v1/d17-1018, arXiv:1707.07045.
[42] M. Joshi, D. Chen, Y. Liu, D.S. Weld, L. Zettlemoyer, O. Levy, SpanBERT: Improving pre-training by representing and predicting spans, Trans. Assoc. Comput. Linguist. 8 (2020) 64–77, http://dx.doi.org/10.1162/tacl_a_00300, URL https://www.aclweb.org/anthology/2020.tacl-1.5.
[43] R. Weischedel, M. Palmer, M. Marcus, E. Hovy, S. Pradhan, L. Ramshaw, N. Xue, A. Taylor, J. Kaufman, M. Franchini, M. El-Bachouti, R. Belvin, A. Houston, OntoNotes Release 5.0 LDC2013T19, Linguistic Data Consortium, 2013.
[44] Dair-Iitd, Open IE 5.1. URL https://github.com/dair-iitd/OpenIE-standalone.
[45] S. Saha, Mausam, Open information extraction from conjunctive sentences, in: Proceedings Ofthe 27th International Conference on Computational Linguistics, 2018.
[46] H. Pal, Mausam, Demonyms and Compound Relational Nouns in Nominal Open IE, 2016, http://dx.doi.org/10.18653/v1/w16-1307.
[47] S. Saha, H. Pal, Mausam, Bootstrapping for numerical open IE, in: ACL 2017 - 55th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference (Long Papers), 2017, http://dx.doi.org/10.18653/v1/P17-2050.
[48] J. Christensen, S. Soderland, O. Etzioni, An analysis of open information extraction based on semantic role labeling categories and subject descriptors, in: Proceeding of K-CAP '11 Proceedings of the Sixth International Conference on Knowledge Capture, 2011.
[49] G. Stanovsky, J. Michael, L. Zettlemoyer, I. Dagan, Supervised open information extraction, in: NAACL-HLT, 2018.
[50] T.-H. Wu, Z. Wu, B. Kao, P. Yin, Towards practical open knowledge base canonicalization, in: Proceedings of the 27th ACM International Conference on Information and Knowledge Management, 2018, pp. 883–892.
[51] L. Galárraga, G. Heitz, K. Murphy, F.M. Suchanek, Canonicalizing open knowledge bases, in: Proceedings of the 23rd Acm International Conference on Conference on Information and Knowledge Management, 2014, pp. 1679–1688.
[52] X. Lin, L. Chen, Canonicalization of open knowledge bases with side information from the source text, in: 2019 IEEE 35th International Conference on Data Engineering (ICDE), 2019, pp. 950–961, http://dx.doi.org/10.1109/ICDE.2019.00089.
[53] C.J. Mungall, C. Torniai, G.V. Gkoutos, S.E. Lewis, M.A. Haendel, Uberon, an integrative multi-species anatomy ontology, Genome Biol. 13 (1) (2012) 1–20.
[54] H.M. Dingerdissen, F. Bastian, K. Vijay-Shanker, M. Robinson-Rechavi, A. Bell, N. Gogate, S. Gupta, E. Holmes, R. Kahsay, J. Keeney, et al., OncoMX: a knowledgebase for exploring cancer biomarkers in the context of related cancer and healthy data, JCO Clin. Cancer Inform. 4 (2020) 210–220.
[55] K. Kolluru, V. Adlakha, S. Aggarwal, S. Chakrabarti, et al., OpenIE6: Iterative grid labeling and coordination analysis for open information extraction, 2020, arXiv preprint arXiv:2010.03147.
[56] K. Kolluru, S. Aggarwal, V. Rathore, S. Chakrabarti, et al., IMoJIE: Iterative memory-based joint open information extraction, 2020, arXiv preprint arXiv:2005.08178.
[57] G. Stanovsky, I. Dagan, Creating a large benchmark for open information extraction, in: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Austin, Texas, 2016, pp. 2300–2305, http://dx.doi.org/10.18653/v1/D16-1252, URL https://aclanthology.org/D16-1252.
[58] Z. Miao, Y. Li, X. Wang, Rotom: A meta-learned data augmentation framework for entity matching, data cleaning, text classification, and beyond, in: Proceedings of the 2021 International Conference on Management of Data, 2021, pp. 1303–1316.